Cryptography and Network Security
UNIT-1

Syllabus:

UNIT I: Classical Encryption Techniques

Introduction: Security attacks, services & mechanisms, Symmetric Cipher Model, Substitution Techniques, Transportation Techniques, Cyber threats and their defense (Phishing Defensive measures, web based attacks, SQL injection & Defense techniques), Buffer overflow & format string vulnerabilities, TCP session hijacking (ARP attacks, route table modification) UDP hijacking.

Objectives: *The Objectives of this unit is to present an overview of the main concepts of cryptography, understand the threats & attacks, understand ethical hacking.*

COMPUTER SECURITY CONCEPTS:

The National Institute of Standards and Technology (NIST) defines the term computer security as follows:

The protection afforded to an automated information system in order to attain the applicable objectives of preserving the integrity, availability, and confidentiality of information system resources (includes hardware, software, firmware, information/ data, and telecommunications).

This definition introduces three key objectives that are at the heart of computer security:

Confidentiality:

This term covers two related concepts:

Data confidentiality: Assures that private or confidential information is not made available or disclosed to unauthorized individuals.

Privacy: Assures that individuals control or influence what information related to them may be collected and stored and by whom and to whom that information may be disclosed.

Integrity:

This term covers two related concepts:

Data integrity: Assures that information and programs are changed only in a specified and authorized manner.

System integrity: Assures that a system performs its intended function in an unaffected manner, free from deliberate or inadvertent unauthorized manipulation of the system.

Availability: Availability of information refers to ensuring that authorized parties are able to access the information when needed.

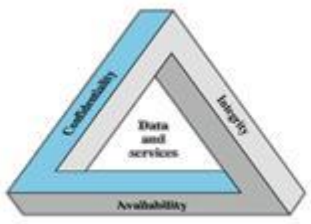


Figure: CIA Triad

THE OSI SECURITY ARCHITECTURE

- The **Open Systems Interconnection (OSI) security architecture provides** a systematic framework for defining security attacks, mechanisms, and services.

BASIC TERMINOLOGY:

Threat: A potential for violation of security, which exists when there is a circumstance, capability, action, or event that could breaks security and cause harm. That is, a threat is a possible danger that might exploit a vulnerability.

Attack: A violation on system security that derives from an intelligent threat; that is, an intelligent act that is a deliberate attempt to evade security services and violate the security policy of a system.

ASPECTS OF SECURITY:

consider 3 aspects of information security:

Security attack: Any action that compromises the security of information owned by an organization.

Security mechanism: A process that is designed to detect, prevent, or recover from a security attack.

Security service: A processing or communication service that enhances the security of the data processing systems and the information transfers of an organization. The services are intended to counter security attacks, and they make use of one or more security mechanisms to provide the service.

<u>SECURITY ATTACKS:</u>

Security attacks are classifieds into two:

- ***Passive attacks and***
- ***Active attacks.***

Apassive attack attempts to learn or make use of information from the system but does not affect system resources.

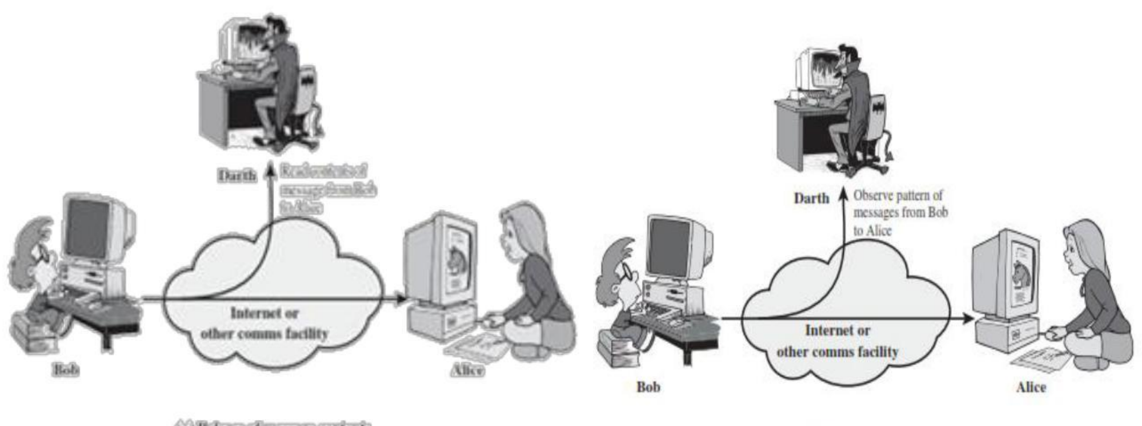
An active attack attempts to alter system resources or affect their operation.

<u>Passive Attacks:</u>

Passive attacks are in the nature of eavesdropping on, or monitoring of, transmissions. The goal of the opponent is to obtain information that is being transmitted.

Two types of passive attacks are the **release of message contents** and **traffic analysis**.

<u>Release of message contents</u>: The **release of message contents is easily understood.** A telephone conversation, an electronic mail message, and a transferred file may contain sensitive or confidential information. We would like to prevent an opponent from learning the contents of these transmissions.


Darth
Read contents of message from Bob to Alice
Internet or other comms facility
Bob
Alice
Darth
Observe pattern of messages from Bob to Alice
Internet or other comms facility
Bob
Alice

(a) Release of message contents (b) Traffic analysis

<u>TRAFFIC ANALYSIS:</u>

Suppose that we had a way of masking the contents of messages or other information traffic so that opponents, even if they captured the message, could not extract the information from the message.

The common technique for masking contents is encryption.

If we had encryption protection in place, an opponent might still be able to observe the pattern of these messages. The opponent could determine the location and identity of communicating hosts and could observe the frequency and length of messages being exchanged.

This information might be useful in guessing the nature of the communication that was taking place.

Passive attacks are very difficult to detect, because they do not involve any alteration of the data.

Typically, the message traffic is sent and received in an apparently normal fashion, and neither the sender nor receiver is aware that a third party has read the messages or observed the traffic pattern.

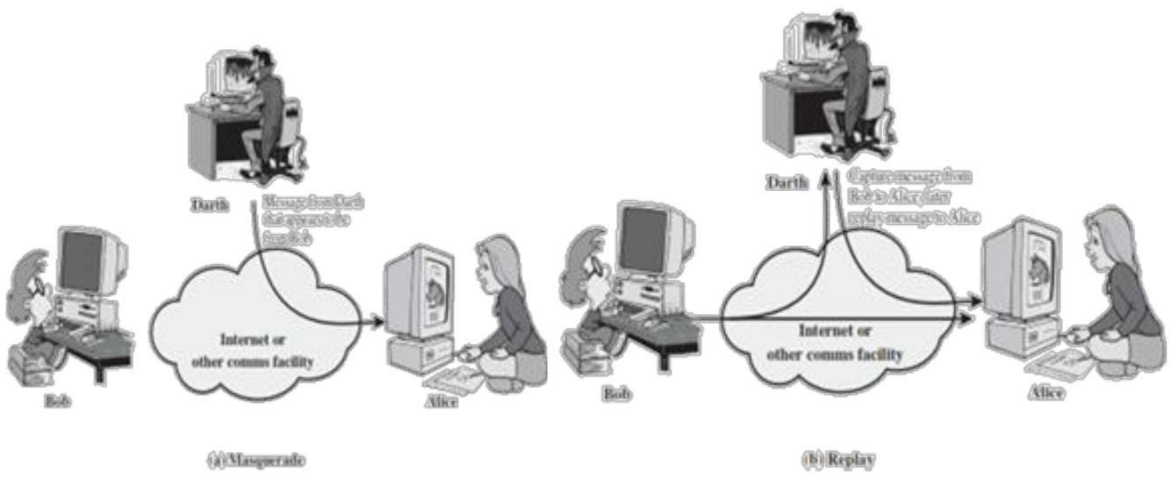
However, it is feasible to prevent the success of these attacks, usually by means of encryption.

Thus, the emphasis in dealing with passive attacks is on prevention rather than detection.

<u>Active Attacks:</u>

Active attacks involve some modification of the data stream or the creation of a false stream and can be subdivided into **four** categories: **masquerade, replay, modification of messages, and denial of service**.

A masquerade attack is an attack that uses a fake identity, to gain unauthorized access to personal computer information through legitimate access identification. For example, authentication sequences can be captured and replayed after a valid authentication sequence has taken place, thus enabling an authorized entity with few privileges to obtain extra privileges by impersonating an entity that has those privileges.

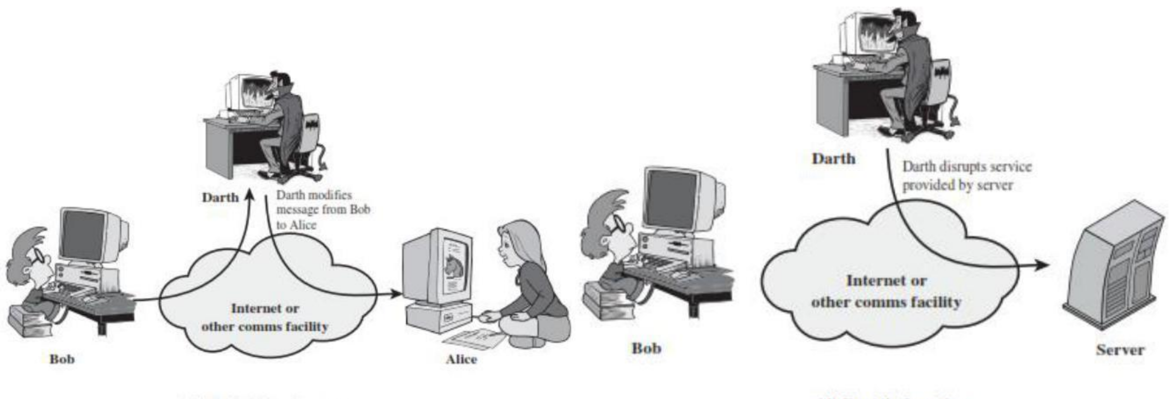

Darth
Message from Darth that appears to be from Bob
Internet or other comms facility
Bob
Alice
Darth
Capture message from Bob to Alice; later replay message to Alice
Internet or other comms facility
Bob
Alice

(a) Masquerade (b) Replay

Replay involves the passive capture of a data unit and its subsequent retransmission to produce an unauthorized effect.

Modification of messages simply means that some portion of a valid message is altered, or that messages are delayed or reordered, to produce an unauthorized effect.

For example, a message meaning "Allow John Smith to read confidential file *accounts" is modified to mean "Allow Fred Brown to read* confidential file *accounts."*


Darth
Darth modifies message from Bob to Alice
Internet or other comms facility
Bob
Alice
Darth
Darth disrupts service provided by server
Internet or other comms facility
Bob
Server

(c) Modification of messages (d) Denial of service

The **denial of service prevents the normal use or management of** communications facilities. This attack may have a specific target; for example, an entity may suppress all messages directed to a particular destination. Another form of service denial is the disruption of an entire network, either by disabling the network or by overloading it with messages so as to degrade performance.

SECURITY SERVICES:

Security service means a processing or communication service that is provided by a system to give a specific kind of protection to system resources.
X.800 divides these services into

- **AUTHENTICATION**
- **ACCESS CONTROL**
- **DATA CONFIDENTIALITY**
- **DATA INTEGRITY**
- **NONREPUDIATION**
- **AVAILABILITY**

AUTHENTICATION:

The authentication service is concerned with assuring that a communication is authentic. In the case of a **single message**, its function of the authentication service is to assure the recipient that the message is from the source that it claims to be from. In the case of an **ongoing interaction**, such as the connection of a terminal to a host, two aspects are involved. First, at the time of connection initiation, the service assures that the two entities are authentic, that is, that each is the entity that it claims to be. Second, the service must assure that the connection is not interfered with in such a way that a third party can masquerade as one of the two legitimate parties for the purposes of unauthorized transmission or reception.

Two specific authentication services are defined

- Peer entity authentication
- Data origin authentication

Peer entity authentication: Provides for the corroboration of the identity of a peer entities involved in communication. It is used for providing authentication at the time of connection establishment and during the process of data transmission.

Data origin authentication: Provides for the corroboration of the source of a data unit. It does not provide protection against the duplication or modification of data units. This type of service supports applications like electronic mail, where there are no prior interactions between the communicating entities **.**

ACCESS CONTROL:

The prevention of unauthorized use of a resources. Access control is the ability to limit and control the access to host systems and applications via communications links. To achieve this, each entity trying to gain access must first be identified, or authenticated, so that access rights can be tailored to the individual.

DATA CONFIDENTIALITY:

Confidentiality is the protection of transmitted data from passive attacks. The protection of data from unauthorized disclosure.

Types of confidentiality:

- **Connection Confidentiality:** The protection of all user data on a connection.
- **Connectionless Confidentiality:** The protection of all user data in a single data block
- **Selective-Field Confidentiality:** The confidentiality of selected fields within the user data on a connection or in a single data block.
- **Traffic-Flow Confidentiality:** The protection of the information that might be derived from observation of traffic flows.

DATA INTEGRITY: The assurance that data received are exactly as sent by an authorized entity (i.e., contain no modification, insertion, deletion, or replay).

Types of integrity

- **Connection Integrity with Recovery**: Provides for the integrity of all user data on a connection and detects any modification, insertion, deletion, or replay of any data within an entire data sequence, with recovery attempted.
- **Connection Integrity without Recovery** as above, but provides only detection without recovery.
- **Selective-Field Connection Integrity** Provides for the integrity of selected fields within the user data of a data block transferred over a connection and takes the form of determination of whether the selected fields have been modified, inserted, deleted, or replayed.
- **Connectionless Integrity** Provides for the integrity of a single connectionless data block and may take the form of detection of data modification. Additionally, a limited form of replay detection may be provided.
- **Selective-Field Connectionless Integrity** Provides for the integrity of selected fields within a single connectionless data block; takes the form of determination of whether the selected fields have been modified.

NONREPUDIATION:

It is assurance that someone cannot deny something. It is a method of guaranteeing message transmission between parties. Provides protection against denial by one of the entities involved in a communication of having participated in all or part of the communication.

- **Nonrepudiation, Origin: Proof** that the message was sent by the specified party.
- **Nonrepudiation, Destination:** Proof that the message was received by the specified party.

AVAILABILITY:

Availability is the method with assure the information and communications will be ready for use when excepted. Information is kept available to authorized persons when they need it. The availability can be significantly affected by a variety of attacks which are susceptible to authentication, encryption etc., whereas some attacks require physical action for preventing and recovering from the loss of availability

SECURITY MECHANISMS:

Security mechanism are categorized into two types. They are,

- **SPECIFIC SECURITY MECHANISMS**
- **PERVASIVE SECURITY MECHANISMS**

SPECIFIC SECURITY MECHANISMS:

These mechanisms are incorporated into the appropriate protocol layer in order to provide some of the OSI security services.

- **Encipherment:** It refers to the process of applying mathematical algorithms to transform data into a form that is not readily intelligible. The transformation and subsequent recovery of the data depend on an algorithm and encryption keys.
- **Digital Signature:** Data appended to, or a cryptographic transformation of, a data unit must preserve the integrity of the data and prevents it from any unauthorized access.
- **Access Control:** A variety of mechanisms that enforce access rights to resources.
- **Data Integrity:** A variety of mechanisms used to assure the integrity of a data unit or stream of data units.
- **Authentication Exchange:** A mechanism intended to ensure the identity of an entity by means of information exchange.
- **Traffic Padding:** The insertion of bits into gaps in a data stream to frustrate traffic analysis attempts.
- **Routing Control:** Enables selection of particular physically secure routes for certain data and allows routing changes, especially when a breach of security is suspected.
- **Notarization:** The use of a trusted third party to assure certain properties of a data exchange.

PERVASIVE SECURITY MECHANISMS:

Mechanisms that are not specific to any particular OSI security service or protocol layer.

- **Trusted Functionality:** That which is perceived to be correct with respect to some criteria.
- **Security Label:** the bounding value of a resource which specifies the security attributes associated with that resource.
- **Event Detection:** Detection of security-relevant events.
- **Security Audit Trail:** Data collected and potentially used to facilitate a security audit, which is an independent review and examination of system records and activities.
- **Security Recovery:** Deals with requests from mechanisms, such as event handling and management functions, and takes recovery actions.

SOME BASIC TERMINOLOGY:

An original message is known as the **plaintext.**

The coded message is called the **ciphertext.**

The process of converting from plaintext to ciphertext is known as **enciphering or encryption.** Restoring the plaintext from the ciphertext is **deciphering** or **decryption.**

The many schemes used for encryption constitute the area of study known as **cryptography.** Such a scheme is known as a **cryptographic system or a cipher.**

Techniques used for deciphering a message without any knowledge of the enciphering details fall into the area of **cryptanalysis.** Cryptanalysis is what the layperson calls "breaking the code."

The areas of cryptography and cryptanalysis together are called **cryptology.**

SYMMETRIC CIPHER MODEL:

Symmetric encryption, also referred to as conventional encryption or single-key encryption

Asymmetric encryption scheme has five ingredients.

- **Plaintext:** This is the original intelligible message or data that is fed into the algorithm as input.
- **Encryption algorithm:** The encryption algorithm performs various substitutions and transformations on the plaintext.
- **Secret key:** The secret key is also input to the encryption algorithm. The key is a value independent of the plaintext and of the algorithm. The algorithm will produce a different output depending on the specific key being used at the time. The exact substitutions and transformations performed by the algorithm depend on the key.
- **Ciphertext:** This is the scrambled message produced as output. It depends on the plaintext and the secret key. For a given message, two different keys will produce two different cipher texts. The ciphertext is an apparently random stream of data and, as it stands, is unintelligible.
- **Decryption algorithm:** This is essentially the encryption algorithm run in reverse. It takes the ciphertext and the secret key and produces the original plaintext.

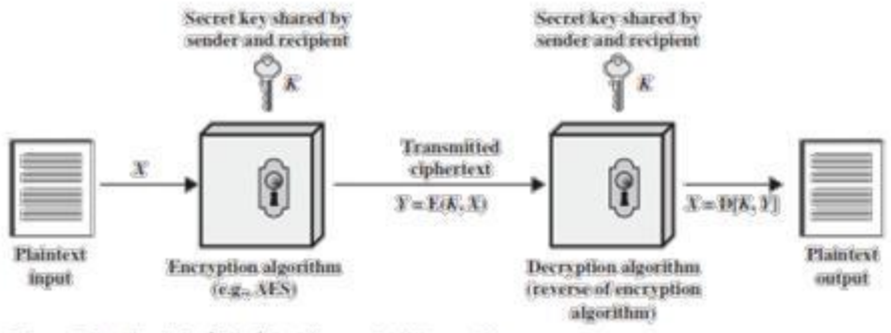

Secret key shared by sender and recipient
K
Secret key shared by sender and recipient
K
X
Transmitted ciphertext
Y = E(K, X)
X = D[K, Y]
Plaintext input
Encryption algorithm (e.g., AES)
Decryption algorithm (reverse of encryption algorithm)
Plaintext output

Figure 2.1 Simplified Model of Symmetric Encryption

Requirements:

There are two requirements for secure use of conventional encryption:

1. We need a strong encryption algorithm
2. a secret key known only to sender / receiver: Sender and receiver must have obtained copies of the secret key in a secure fashion and must keep the key secure

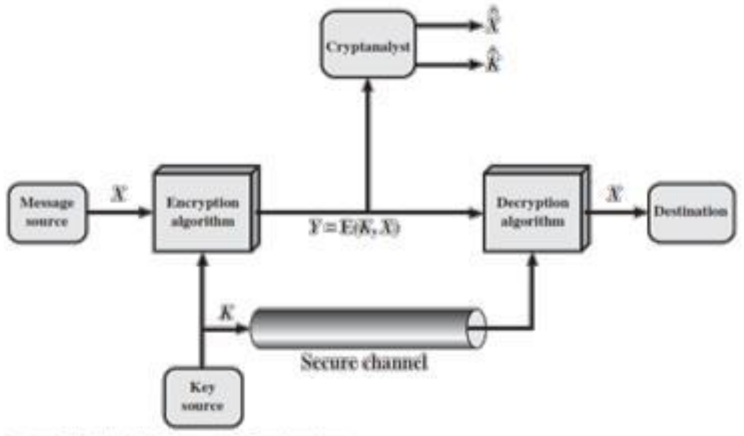


Figure 2.2 Model of Symmetric Cryptosystem

The essential elements of a symmetric encryption scheme, in the Figure. A source produces a message in plaintext, $X = [X_1, X_2 \ldots\ldots X_M]$. The elements of are letters in some finite alphabet. Traditionally, the alphabet usually consisted of the 26 capital letters. Nowadays, the binary alphabet {0, 1} is typically used. For encryption, a key of the form $K = [K_1, K_2 \ldots K_J]$is generated. If the key is generated at the message source, then it must also be provided to the destination by means of some secure channel. Alternatively, a third party could generate the key and securely deliver it to both source and destination. With the message and the encryption key as input, the encryption algorithm forms the ciphertext $Y=[Y_1, Y_2 \ldots Y_N]$. We can write this as $Y=E(K,X)$. This notation indicates that is produced by using encryption algorithm E as a function of the plaintext *X*, with the specific function determined by the value of the key *K*. The intended receiver, in possession of the key, is able to invert the transformation: X=D(K,Y). An opponent, observing Y but not having access to *K* or *X* , may attempt to recover *X* or *K* or both *X* and *K*. It is assumed that the opponent knows the encryption (E) and decryption (D) algorithms. If the opponent is interested in only this particular message, then the focus of the effort is to recover *X* by generating a plaintext estimate X∧. Often, however, the opponent is interested in being able to read future messages as well, in which case an attempt is made to recover by generating an estimate k .

Cryptography:

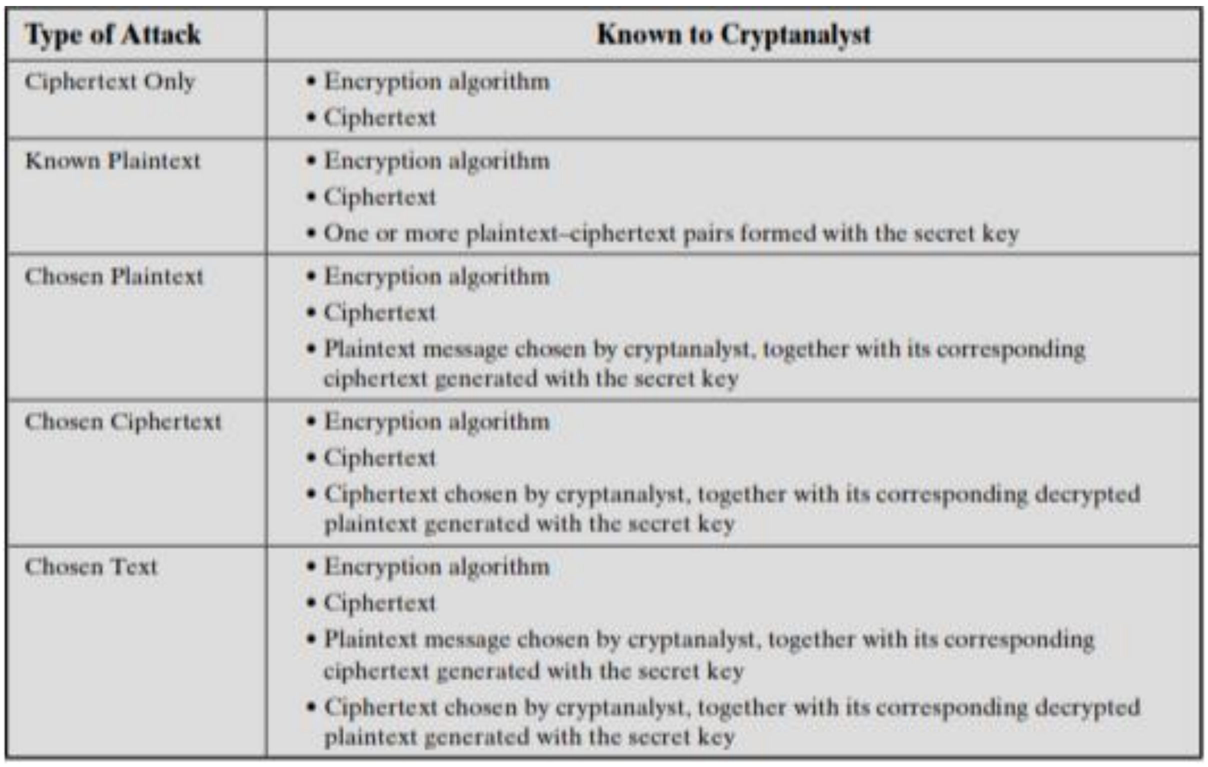
Cryptographic systems are characterized along three independent dimensions:

1. **The type of operations used for transforming plaintext to ciphertext.** All encryption algorithms are based on two general principles: substitution, in which each element in the plaintext (bit, letter, group of bits or letters) is mapped into another element, and transposition, in which elements in the plaintext are rearranged. The fundamental requirement is that no information be lost (that is, that all operations are reversible). Most systems, referred to as *product systems, involve multiple stages of substitutions and* transpositions.
2. **The number of keys used.** If both sender and receiver use the same key, the system is referred to as symmetric, single-key, secret-key, or conventional encryption. If the sender and receiver use different keys, the system is referred to as asymmetric, two-key, or public-key encryption.
3. **The way in which the plaintext is processed**. A *block cipher* processes the input one block of elements at a time, producing an output block for each input block. A *stream cipher processes the input elements continuously*, producing output one element at a time, as it goes along.

Cryptanalysis:

There are two general approaches to attacking a conventional encryption scheme:

- **Cryptanalysis:** Cryptanalytic attacks rely on the nature of the algorithm plus perhaps some knowledge of the general characteristics of the plaintext or even some sample plaintext ciphertext pairs. This type of attack exploits the characteristics of the algorithm to attempt to deduce a specific plaintext or to deduce the key being used.
- **Brute-force attack:** The attacker tries every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained. On average, half of all possible keys must be tried to achieve success.



Type of Attack	Known to Cryptanalyst
Ciphertext Only	• Encryption algorithm • Ciphertext
Known Plaintext	• Encryption algorithm • Ciphertext • One or more plaintext–ciphertext pairs formed with the secret key
Chosen Plaintext	• Encryption algorithm • Ciphertext • Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key
Chosen Ciphertext	• Encryption algorithm • Ciphertext • Ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key
Chosen Text	• Encryption algorithm • Ciphertext • Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key • Ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key

- An encryption scheme is **unconditionally Secure** if the ciphertext generated by the scheme does not contain enough information to determine uniquely the corresponding plaintext, no matter how much ciphertext is available. That is, no matter how much time an opponent has, it is impossible for him or her to decrypt the ciphertext simply because the required information is not there.
- An encryption scheme: **computationally secure** if The cost of breaking the cipher exceeds the value of information and the time required to break the cipher exceeds the lifetime of information

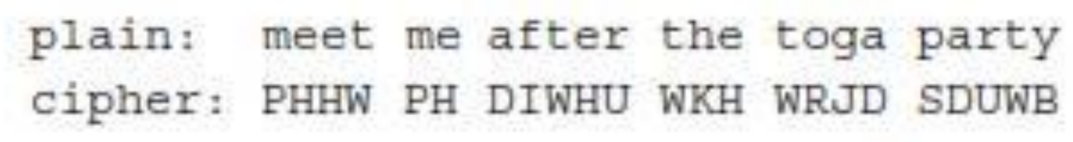
SUBSTITUTION TECHNIQUES:

The two basic building blocks of all encryption techniques are substitution and transposition.

- A substitution technique is one in which the letters of plaintext are replaced by other letters or by numbers or symbols. If the plaintext is viewed as a sequence of bits, then substitution involves replacing plaintext bit patterns with ciphertext bit patterns.

Caesar Cipher:

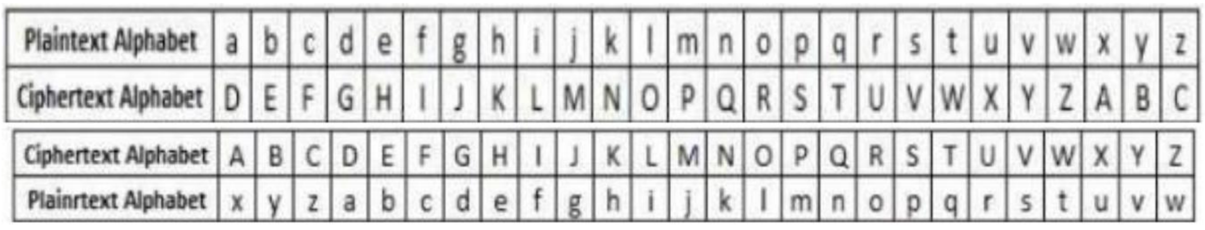
The earliest known, and the simplest, use of a substitution cipher was by Julius Caesar. The Caesar cipher involves replacing each letter of the alphabet with the letter standing three places further down the alphabet. For example,



```
plain:  meet me after the toga party
cipher: PHHW PH DIWHU WKH WRJD SDUWB
```

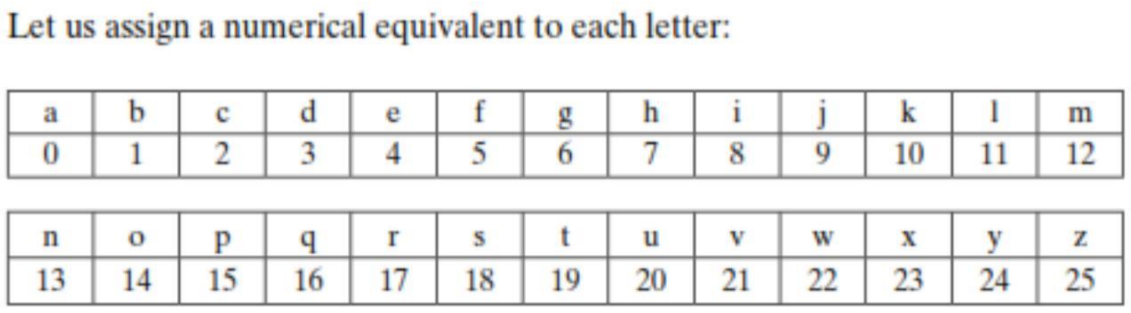
Process of Caesar Cipher:

- In order to encrypt a plaintext letter, the sender positions the sliding ruler underneath the first set of plaintext letters and slides it to LEFT by the number of positions of the secret shift(here 3).
- The plaintext letter is then encrypted to the ciphertext letter on the sliding ruler underneath. The result of this process is depicted in the following illustration for an agreed shift of three positions.



Plaintext Alphabet	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Ciphertext Alphabet	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Ciphertext Alphabet	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Plainrtext Alphabet	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w



Let us assign a numerical equivalent to each letter:

a	b	c	d	e	f	g	h	i	j	k	l	m
0	1	2	3	4	5	6	7	8	9	10	11	12

n	o	p	q	r	s	t	u	v	w	x	y	z
13	14	15	16	17	18	19	20	21	22	23	24	25

Then the algorithm can be expressed as follows. For each plaintext letter *p, substitute* the ciphertext letter *C*

$$C=E(3,p)=(p+3) \bmod 26$$

Ashift may be of any amount, so that the general Caesar algorithm is

$$C = E(k, p) = (p + k) \bmod 26$$

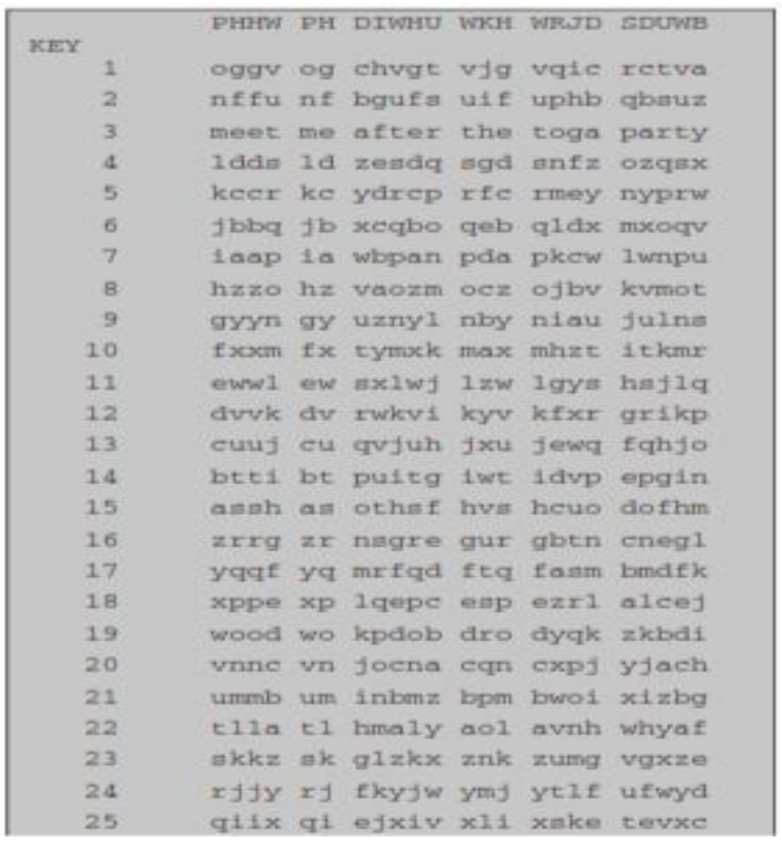
where *k takes on a value in the range 1 to 25. The decryption algorithm is simply*

$$p = D(k, C) = (C - k) \bmod 26$$

If it is known that a given ciphertext is a Caesar cipher, then a brute-force cryptanalysis is easily performed: simply try all the 25 possible keys. Following figure shows the results of applying this strategy to the example ciphertext. In this case, the plaintext leaps out as occupying the third line.

Three important characteristics of this problem enabled us to use a brute force cryptanalysis:

1. **The encryption and decryption algorithms are known.**
2. **There are only 25 keys to try.**
3. **The language of the plaintext is known and easily recognizable.**



```
          PHHW PH DIWHU WKH WRJD SDUWB
KEY
  1       oggv og chvgt vjg vqic rctva
  2       nffu nf bgufs uif uphb qbsuz
  3       meet me after the toga party
  4       ldds ld zesdq sgd snfz ozqsx
  5       kccr kc ydrcp rfc rmey nyprw
  6       jbbq jb xcqbo qeb qldx mxoqv
  7       iaap ia wbpan pda pkcw lwnpu
  8       hzzo hz vaozm ocz ojbv kvmot
  9       gyyn gy uznyl nby niau julns
 10       fxxm fx tymxk max mhzt itkmr
 11       ewwl ew sxlwj lzw lgys hsjlq
 12       dvvk dv rwkvi kyv kfxr grikp
 13       cuuj cu qvjuh jxu jewq fqhjo
 14       btti bt puitg iwt idvp epgin
 15       assh as othsf hvs hcuo dofhm
 16       zrrg zr nsgre gur gbtn cnegl
 17       yqqf yq mrfqd ftq fasm bmdfk
 18       xppe xp lqepc esp ezrl alcej
 19       wood wo kpdob dro dyqk zkbdi
 20       vnnc vn jocna cqn cxpj yjach
 21       ummb um inbmz bpm bwoi xizbg
 22       tlla tl hmaly aol avnh whyaf
 23       skkz sk glzkx znk zumg vgxze
 24       rjjy rj fkyjw ymj ytlf ufwyd
 25       qiix qi ejxiv xli xske tevxc
```

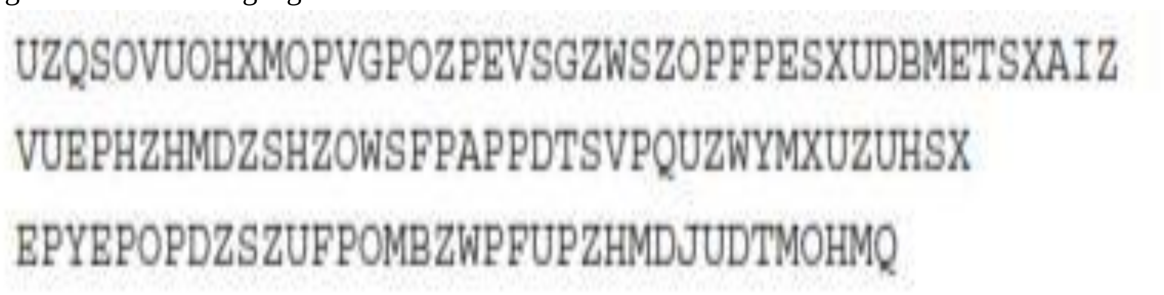
MONOALPHABETIC CIPHERS:

- Monoalphabetic cipher is a substitution cipher in which for a given key, the cipher alphabet for each plain alphabet is fixed throughout the encryption process. With only 25 possible keys, the Caesar cipher is far from secure. A dramatic increase in the key space can be achieved by allowing an arbitrary substitution. Before proceeding, we define the term *permutation.* A permutation of a finite set of elements *S* is an ordered sequence of all the elements of *S*, with each element appearing exactly once.

  For example, if *S* = {a, b, c}, there are six permutations of *S:*

  abc, acb, bac, bca, cab, cba

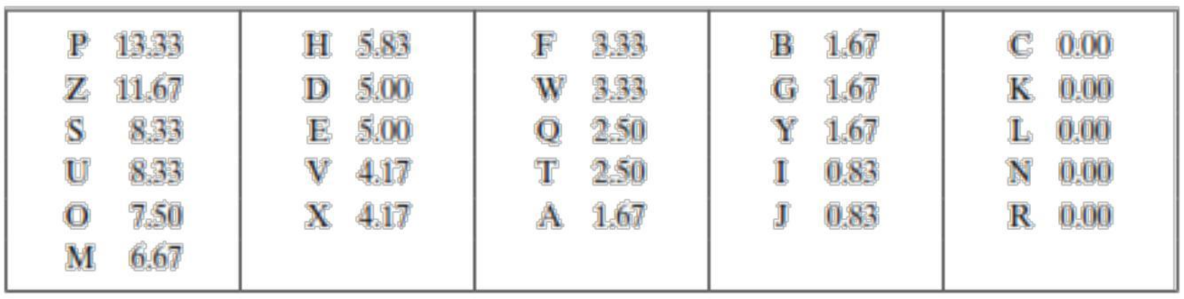
If the cryptanalyst knows the nature of the plaintext, then the analyst can exploit the regularities of the language.



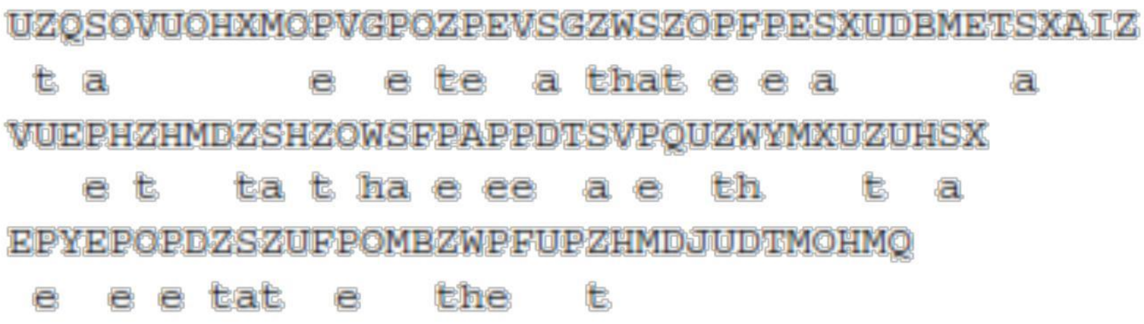
```
UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAIZ

VUEPHZHMDZSHZOWSFPAPPDTSVPQUZWYMXUZUHSX

EPYEPOPDZSZUFPOMBZWPFUPZHMDJUDTMOHMQ
```

As a first step, the relative frequency of the letters can be determined and compared to a standard frequency distribution for English, such as is shown in Figure. If the message were long enough, this technique alone might be sufficient, but because this is a relatively short message, we cannot expect an exact match. A powerful tool is to look at the frequency of two-letter combinations, known as **digrams.**

The following table shows the frequency of letters in the above sentences



P	13.33	H	5.83	F	3.33	B	1.67	C	0.00
Z	11.67	D	5.00	W	3.33	G	1.67	K	0.00
S	8.33	E	5.00	Q	2.50	Y	1.67	L	0.00
U	8.33	V	4.17	T	2.50	I	0.83	N	0.00
O	7.50	X	4.17	A	1.67	J	0.83	R	0.00
M	6.67								

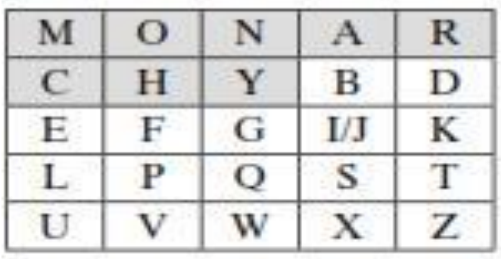


```
UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAIZ
 t a          e   e te  a that e e a        a
VUEPHZHMDZSHZOWSFPAPPDTSVPQUZWYMXUZUHSX
   e t   ta t ha e ee  a e  th     t  a
EPYEPOPDZSZUFPOMBZWPFUPZHMDJUDTMOHMQ
 e  e e tat  e   the    t
```

Monoalphabetic ciphers are easy to break because they reflect the frequency data of the original alphabet.

Playfair Cipher:

The best-known multiple-letter encryption cipher is the Playfair, which treats digrams in the plaintext as single units and translates these units into ciphertext digrams. The Playfair algorithm is based on the use of a 5 * 5 matrix of letters constructed using a keyword.



M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

In this case, the keyword is *monarchy*. The matrix is constructed by filling in the letters of the keyword (minus duplicates) from left to right and from top to bottom, and then filling in the remainder of the matrix with the remaining letters in alphabetic order. The letters I and J count as one letter. Plaintext is encrypted two letters at a time, according to the following rules:

1. **Repeating plaintext letters that are in the same pair are separated with a filler** letter, such as x, so that balloon would be treated as ba lx lo on.
2. **Two plaintext letters that fall in the same row of the matrix are each replace** by the letter to the right, with the first element of the row circularly following the last. For example, ar is encrypted as RM.
3. **Two plaintext letters that fall in the same column are each replaced by the** letter beneath, with the top element of the column circularly following the last. For example, mu is encrypted as CM.

4. **Otherwise, each plaintext letter in a pair is replaced by the letter that lies in** its own row and the column occupied by the other plaintext letter. Thus, hs becomes BP and ea becomes IM (or JM, as the encipherer wishes).

The Playfair cipher is a great advance over simple monoalphabetic ciphers. For one thing, whereas there are only 26 letters, there are 26 * 26 = 676 digrams, so that identification of individual digrams is more difficult. Despite this level of confidence in its security, the Playfair cipher is relatively easy to break, because it still leaves much of the structure of the plaintext language intact. A few hundred letters of ciphertext are generally sufficient.

Hill Cipher:

Another interesting multiletter cipher is the Hill cipher, developed by the mathematician Lester Hill in 1929. This encryption algorithm takes *m successive plaintext letters* and substitutes for them *m* ciphertext letters. The substitution is determined by *m linear equations in which each character is assigned a numerical value* (a = 0, b = 1, c, z = 25).

For *m = 3, the system can be described as*

$$c_1 = (k_{11}p_1 + k_{21}p_2 + k_{31}p_3) \bmod 26$$
$$c_2 = (k_{12}p_1 + k_{22}p_2 + k_{32}p_3) \bmod 26$$
$$c_3 = (k_{13}p_1 + k_{23}p_2 + k_{33}p_3) \bmod 26$$

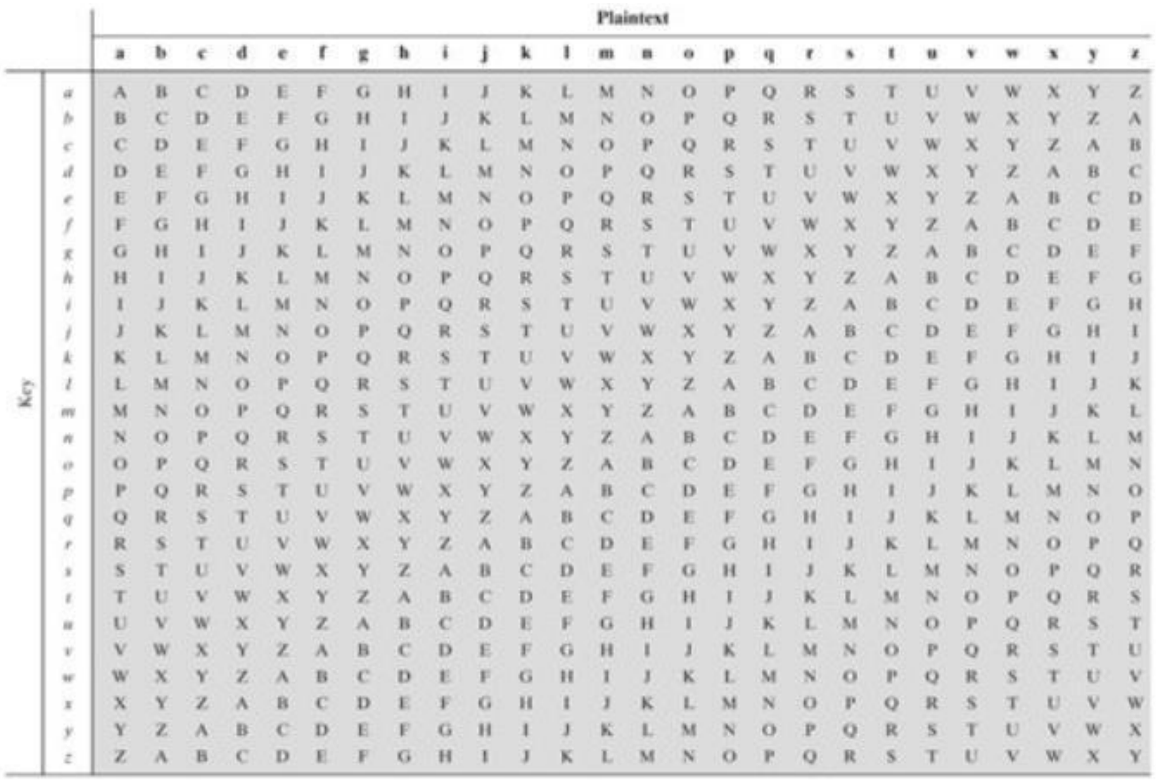
This can be expressed in terms of row vectors and matrices:[6]

$$(c_1\ c_2\ c_3) = (p_1\ p_2\ p_3)\begin{pmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{pmatrix} \bmod 26$$

$$\mathbf{C} = \mathbf{PK} \bmod 26$$

Polyalphabetic Ciphers:

- Another way to improve on the simple monoalphabetic technique is to use different monoalphabetic substitutions as one proceeds through the plaintext message. The general name for this approach is **polyalphabetic substitution cipher.**
- The best known, and one of the simplest, such algorithm is referred to as the Vigenère cipher. In this scheme, the set of related monoalphabetic substitution rules consists of the 26 Caesar ciphers, with shifts of 0 through 25.



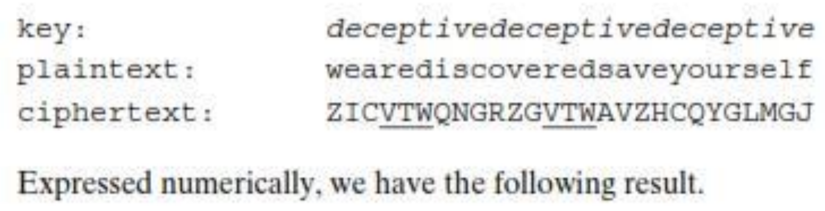
	Plaintext																									
Key	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
b	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
c	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
d	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
e	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
f	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
g	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
h	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
i	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
j	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
k	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
l	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
m	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
n	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
o	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
p	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
r	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
s	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
t	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
u	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
v	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
w	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
x	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Table. The Modern Vigenère Tableau

key: *deceptive*

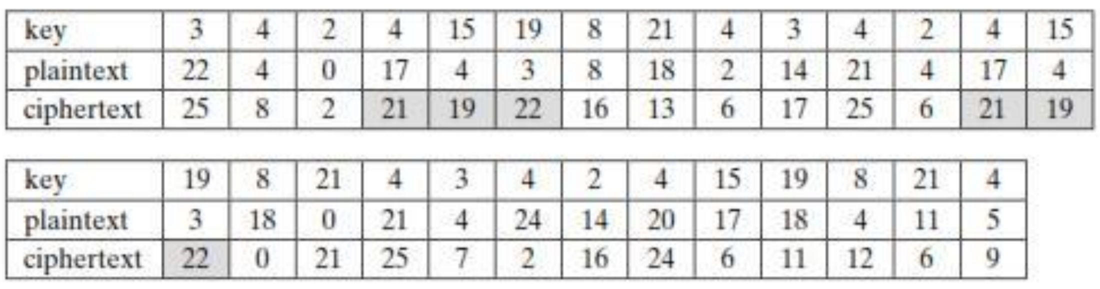
plaintext: wearediscoveredsaveyourself

ciphertext: ZIC**VTW**QNGRZG**VTW**AVZHCQYGLMGJ



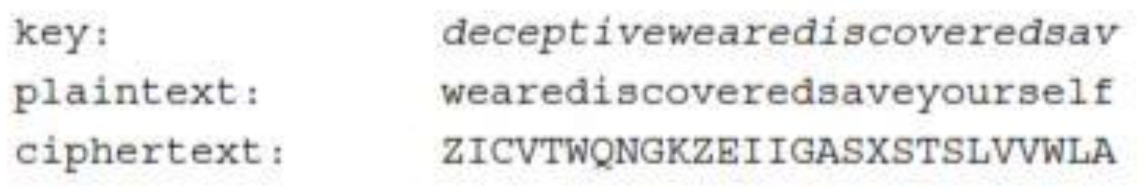
```
key:            deceptivedeceptivedeceptive
plaintext:      wearediscoveredsaveyourself
ciphertext:     ZICVTWQNGRZGVTWAVZHCQYGLMGJ
```

Expressed numerically, we have the following result.



key	3	4	2	4	15	19	8	21	4	3	4	2	4	15	19	8	21	4	3	4	2	4	15	19	8	21	4
plaintext	22	4	0	17	4	3	8	18	2	14	21	4	17	4	3	18	0	21	4	24	14	20	17	18	4	11	5
ciphertext	25	8	2	21	19	22	16	13	6	17	25	6	21	19	22	0	21	25	7	2	16	24	6	11	12	6	9

The periodic nature of the keyword can be eliminated by using a nonrepeating keyword that is as long as the message itself. Vigenère proposed what is referred to as an **autokey system, in which a keyword is concatenated with the plaintext itself to** provide a running key.



```
key:            deceptivewearediscoveredsav
plaintext:      wearediscoveredsaveyourself
ciphertext:     ZICVTWQNGKZEIIGASXSTSLVVWLA
```

Vernam Cipher:

The ultimate defense against such a cryptanalysis is to choose a keyword that is as long as the plaintext and has no statistical relationship to it. Such a system was introduced by an AT&T engineer named Gilbert Vernam in 1918.

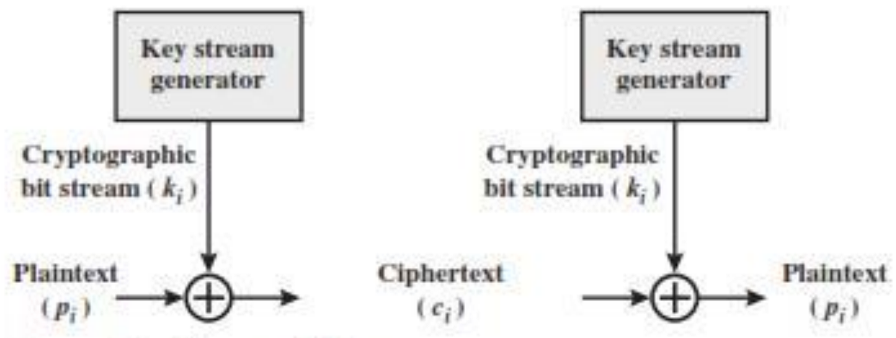


Figure 2.7 Vernam Cipher

<u>One-Time Pad:</u>

Joseph Mauborgne, proposed an improvement to the Vernam cipher that yields the ultimate in security. Mauborgne suggested using a random key that is as long as the message, so that the key need not be repeated. In addition, the key is to be used to encrypt and decrypt a single message, and then is discarded. Each new message requires a new key of the same length as the new message. Such a scheme, known as a **one-time pad, is unbreakable.** It produces random output that bears no statistical relationship to the plaintext. Because the ciphertext contains no information whatsoever about the plaintext, there is simply no way to break the code.

An example should illustrate our point. Suppose that we are using a Vigenère scheme with 27 characters in which the twenty-seventh character is the space character, but with a one-time key that is as long as the message. Consider the ciphertext

ANKYODKYUREPFJBYOJDSPLREYIUNOFDOIUERFPLUYTS

We now show two different decryptions using two different keys:

```
ciphertext: ANKYODKYUREPFJBYOJDSPLREYIUNOFDOIUERFPLUYTS
key:        pxlmvmsydofuyrvzwc tnlebnecvgdupahfzzlmnyih
plaintext:  mr mustard with the candlestick in the hall

ciphertext: ANKYODKYUREPFJBYOJDSPLREYIUNOFDOIUERFPLUYTS
key:        mfugpmiydgaxgoufhklllmhsqdqogtewbqfgyovuhwt
plaintext:  miss scarlet with the knife in the library
```

Suppose that a cryptanalyst had managed to find these two keys. Two plausible plaintexts are produced. How is the cryptanalyst to decide which is the correct decryption (i.e., which is the correct key)? If the actual key were produced in a truly random fashion, then the cryptanalyst cannot say that one of these two keys is more likely than the other. Thus, there is no way to decide which key is correct and therefore which plaintext is correct.

In fact, given any plaintext of equal length to the ciphertext, there is a key that produces that plaintext. Therefore, if you did an exhaustive search of all possible keys, you would end up with many legible plaintexts, with no way of knowing which was the intended plaintext. Therefore, the code is unbreakable.

The one-time pad offers complete security but, in practice, has two fundamental difficulties:

1. **There is the practical problem of making large quantities of random keys.** Any heavily used system might require millions of random characters on a regular basis. Supplying truly random characters in this volume is a significant task.
2. **Even more daunting is the problem of key distribution and protection.** For every message to be sent, a key of equal length is needed by both sender and receiver. Thus, a mammoth key distribution problem exists.

Because of these difficulties, the one-time pad is of limited utility and is useful primarily for low-bandwidth channels requiring very high security.

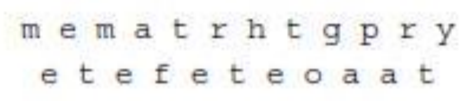
Transposition Techniques:

A very different kind of mapping is achieved by performing some sort of permutation on the plaintext letters. This technique is referred to as a transposition cipher.

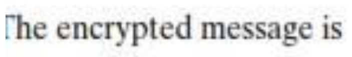
RAIL FENCE TECHNIQUE:

The simplest such cipher is the **rail fence technique, in which the plaintext is** written down as a sequence of diagonals and then read off as a sequence of rows.

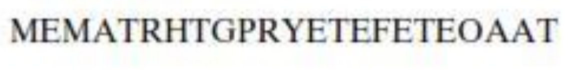
For example, to encipher the message "meet me after the toga party" with a rail fence of depth 2 we write the following:



```
m e m a t r h t g p r y
 e t e f e t e o a a t
```



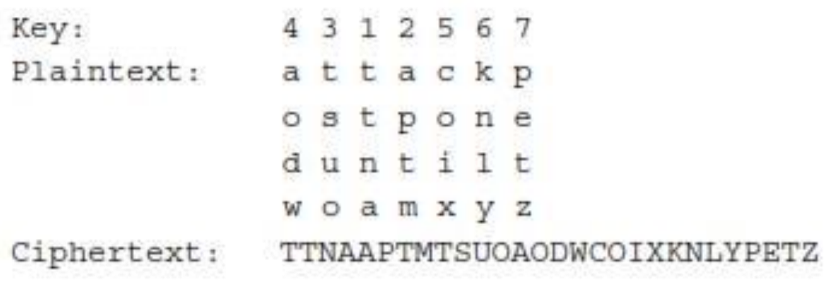
The encrypted message is



MEMATRHTGPRYETEFETEOAAT

Row Transposition Ciphers:

A more complex scheme is to write the message in a rectangle, row by row, and read the message off, column by column, but permute the order of the columns. The order of the columns then becomes the key to the algorithm.



```
Key:          4 3 1 2 5 6 7
Plaintext:    a t t a c k p
              o s t p o n e
              d u n t i l t
              w o a m x y z
Ciphertext:   TTNAAPTMTSUOAODWCOIXKNLYPETZ
```

PHISHING DEFENSIVE MEASURES:

- Phishing is a fraudulent process, which attempts to acquire sensitive information, such as usernames, passwords, and credit card numbers by masquerading as a trustworthy entity in an electronic communication.
- **Spear-phishing emails** have a high success rate because they mimic messages from an authoritative source, such as a financial institution, a communications company, or some other easily recognizable entity with a reputable brand.
- In general, these phishing techniques are manifested in social engineering, URL/Link manipulation, filter evasion e.g., using images to hide malicious links, and website forgery.
- **Web Forgery**, also known as Phishing, is a form of identity theft that occurs when a malicious website impersonates a valid one in order to obtain someone's sensitive information.
- Pharming is another technique intended to redirect a website's traffic to another, fake site.
- Pharming can be conducted either by changing the hosts file on a victim's computer or by exploitation of a vulnerability in DNS server software.
- DNS servers are computers responsible for resolving Internet names into their real IP addresses.
- Compromised DNS servers are sometimes referred to as "poisoned".
- Pharming requires unprotected access to target a computer, such as altering a customer's home computer, rather than a corporate business.
- Avalanche is the name given to the world's most prolific phishing gang, and to the infrastructure it uses to host phishing sites.

SAFE BROWSING TOOL:

Since the web is the most frequently used attack vector, it is important to have protection for browsers, especially when a search is used.

- **The Web of Trust (WOT) Plugin for Safe Browsing:** The WOT is a community-based collection of websites, based on a reputation achieved through the ratings of millions of users. It is a free safe surfing plugin for major browsers and provides website ratings and reviews to help web users as they search, surf and shop online.
- WOT uses color-coded symbols to show the reputation of a site: Green indicates the site is trusted by the community, yellow warns a user to be cautious and red indicates potential

danger. A gray symbol with a question mark means that there is no rating due to a lack of sufficient data. Figure(a) shows that WOT provides a safe rating for each website in the search.

UNIFORM RESOURCE LOCATOR (URL) FILTERING:

URL filters check hyperlinks and URL for specific commands, keywords, and malicious code. This type of filtering is usually utilized by web and email scanning engines. Both Internet Explorer (IE), Chrome, and Firefox provide phishing filters. Phishing and malware protection is accomplished by checking the site that is being visited against lists of reported phishing and malware sites. These lists are automatically downloaded and updated by browsers. So when the Phishing and Malware Protection features are enabled, browsers can provide warnings.

The Location of a List of Phishing Sites:

PhishTank (http://www.phishtank.com/) is a collaborative clearing house for data and information about phishing on the Internet. One can also query or browse this phishing site list.

The Configurations of Phishing Protection Features Employed in Firefox and Internet Explorer (IE):

Firefox provides options for security by checking the two items in the green box. When installing Firefox, these options are enabled by default. The IE8 configuration is by checking the SmartScreen in the Advanced Tab of Internet Options. SmartScreen in enabled by default during the installation process.

The Manner in Which Phishing Site Warnings Are Displayed in IE and Firefox:

IE7 not only clearly labels it as such in the red area at the top of the page, but in addition indicating that HTTPS is not used. A site like Paypal would definitely have a secure site. Unfortunately, it is probably too late for an individual that reaches this point, since the malicious scripts will undoubtedly be loaded into their machine when the site is accessed. IE8 provides a clear warning on the screen.

The Use of a Browser Filter to Block a Phishing Site:

Since a browser may not be able to download its phishing site list in time, a phishing/ malware site may still evade the filtering process. A user should always take precautions, since a phishing websi te may emerge any moment and in this situation the browser filter is always an afterthought.

THE OBFUSCATED URL AND THE REDIRECTION TECHNIQUE:

Two of the most common techniques employed in phishing are the confusing/obfuscated URL and the redirection technique.

For example, the following URLs appear to be an ebay site since ebay is prominently displayed in the listing.

http://ebay.hut2.ru

The other technique is redirection, which is illustrated in the following URL:

http://www.paypal.com/url.php?url = “http://phishing.com”

In this case Paypal appears to be the site, but then it is redirected to phishing.com. This latter technique is an effective phishing approach, since it appears that a legitimate site is being visited while, in fact, redirection to a phishing site is actually taking place.

WEB-BASED ATTACKS:

- The vulnerabilities in web-based attacks are manifested in a variety of ways. For example, the inadequate validation of user input may occur in one of the following attacks: **Cross-Site Scripting (XSS or CSS), HTTP Response Splitting or SQL Injection**.

HTTP RESPONSE SPLITTING ATTACKS:

- HTTP response splitting occurs when:
    - Data enters a web application through an untrusted source, most frequently an HTTP request.
    - The data is included in an HTTP response header sent to a web user without being validated for malicious characters.
- At its root, the attack is straightforward: an attacker passes malicious data to a vulnerable application, and the application includes the data in an HTTP response header.
- HTTP response splitting attacks may happen where the server script embeds user data in HTTP response headers without appropriate sanitation.
- This typically happens when the script embeds user data in the redirection URL of a redirection response (HTTP status code 3xx), or when the script embeds user data in a cookie value or name when the response sets a cookie.
- Attacker uses a web server, which has a vulnerability enabling HTTP response splitting, and a proxy/cache server in a HTTP response splitting attack.
- HTTP response splitting is the attacker’s ability to send a single HTTP request that forces the web server to form an output stream, which is then interpreted by the target as two HTTP responses instead of one response.

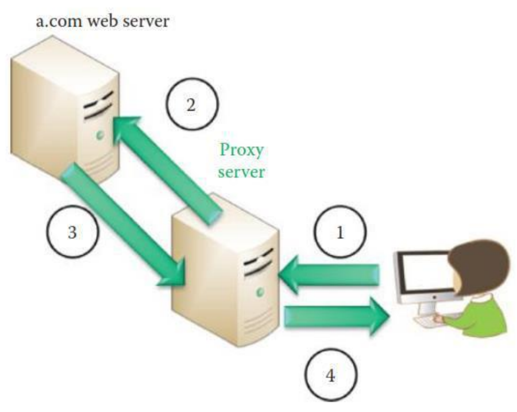


FIGURE: A normal operation for executing a Redirect Script for language preference and the response is cached in a proxy server.

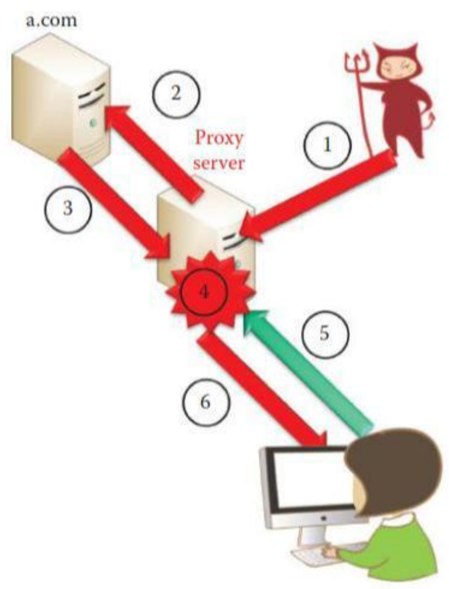


FIGURE: Attacker uses a.com web server, which has a vulnerability enabling HTTP response splitting, and a proxy/cache server in a HTTP response splitting attack. A victim will retrieve the cached second response when accessing the a.com.

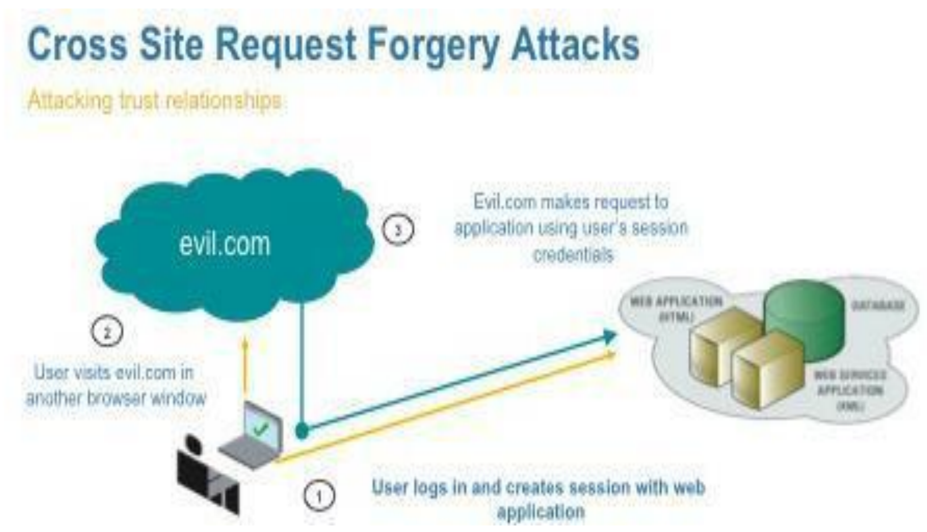
Steps

1. An attacker sends two HTTP requests to the proxy server.
2. The proxy server forwards two HTTP requests to the a.com web server.
3. The a.com web server sends back one HTTP response to each request and the proxy only accepts the first response message.
4. The proxy server interprets the accepted response as two HTTP response messages
    1. The first request is matched to the first response. A first HTTP response, which is a 302 (redirection) response.
    2. The second request (http://a.com/index.html) is matched to the second response. A second HTTP response, which is a 200 response, has a content comprised of 26 bytes of HTML.
5. A victim sends a request to http://a.com/index.html.
6. The victim receives the second response message. The problem is that the content in the second response can be any script that will be executed by the browser.

CROSS-SITE REQUEST FORGERY (CSRF OR XSRF):

Cross-Site Request Forgery (CSRF) is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. CSRF attacks specifically target state-changing requests, not theft of data, since the attacker has no way to see the response to the forged request. With a little help of social engineering (such as sending a link via email or chat), an attacker may trick the users of a web application into executing actions of the attacker's choosing. If the victim is a normal user, a successful CSRF attack can force the user to perform state changing requests like transferring funds, changing their email address, and so forth. If the victim is an administrative account, CSRF can compromise the entire web application.

Cookies are small files which are stored on a user's computer. They are designed to hold a modest amount of data specific to a particular client and website, and can be accessed either by the web server or the client computer.


Cross Site Request Forgery Attacks
Attacking trust relationships
evil.com
3
Evil.com makes request to application using user's session credentials
2
User visits evil.com in another browser window
WEB APPLICATION (HTML)
DATABASE
WEB SERVICES APPLICATION (XML)
1
User logs in and creates session with web application

<u>CROSS-SITE SCRIPTING (XSS) ATTACKS:</u>

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into trusted web sites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it. An attacker can use XSS to send a malicious script to an unsuspecting user. The end user's browser has no way to know that the script should not be trusted, and will execute the script. Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site. These scripts can even rewrite the content of the HTML page.

<u>NON-PERSISTENT XSS ATTACKS:</u>

The *non-persistent* (or *reflected*) cross-site scripting vulnerability is by far the most common type. These holes show up when the data provided by a web client, most commonly in HTTP query parameters (e.g. HTML form submission), is used immediately by server-side scripts to parse and display a page of results for and to that user, without properly sanitizing the request.

<u>PERSISTENT XSS ATTACKS:</u>

The *persistent* (or *stored*) XSS vulnerability is a more disturbing variant of a cross-site scripting flaw: it occurs when the data provided by the attacker is saved by the server, and then permanently displayed on "normal" pages returned to other users in the course of regular browsing, without proper HTML escaping.

Aclassic example of this is with online message boards where users are allowed to post HTML formatted messages for other users to read.

<u>DATABASE DEFENSIVE MEASURES:</u>

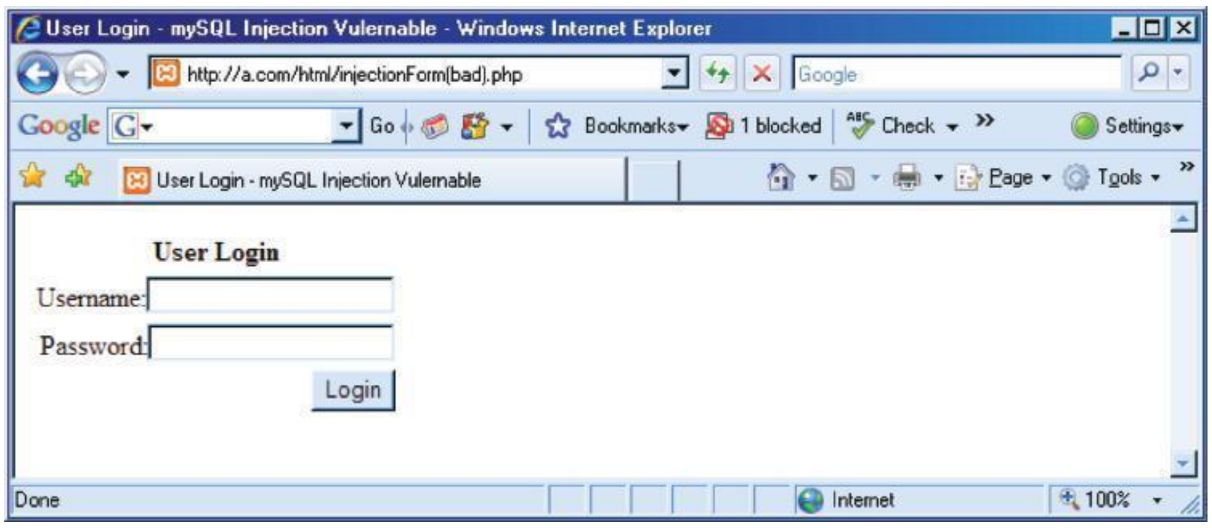
<u>STRUCTURED QUERY LANGUAGE (SQL) INJECTION ATTACKS:</u>

ASQL injection attack consists of insertion or "injection" of a SQL query via the input data from the client to the application. A successful SQL injection exploit can read sensitive data from the database, modify database data (Insert/Update/Delete), execute administration operations on the database (such as shutdown the DBMS), recover the content of a given file present on the DBMS file system and in some cases issue commands to the operating system. SQL injection attacks are a type of injection attack, in which SQL commands are injected into data-plane input in order to affect the execution of predefined SQL commands.

<u>The Manner in Which to Execute a SQL Injection Attack:</u>

As an example of a SQL injection attack, consider the normal user login request shown in Figure (a). A user supplies their username and password, and this SQL query checks to see if the user/password combination is in the database. The query is of the form

$query = "SELECT username,password FROM login WHERE username = '$username' AND password = '$password'";


User Login - mySQL Injection Vulernable - Windows Internet Explorer
http://a.com/html/injectionForm(bad).php
Google
Bookmarks
1 blocked
Check
Settings
User Login - mySQL Injection Vulernable
Page
Tools
Done
Internet
100%

User Login

Username:

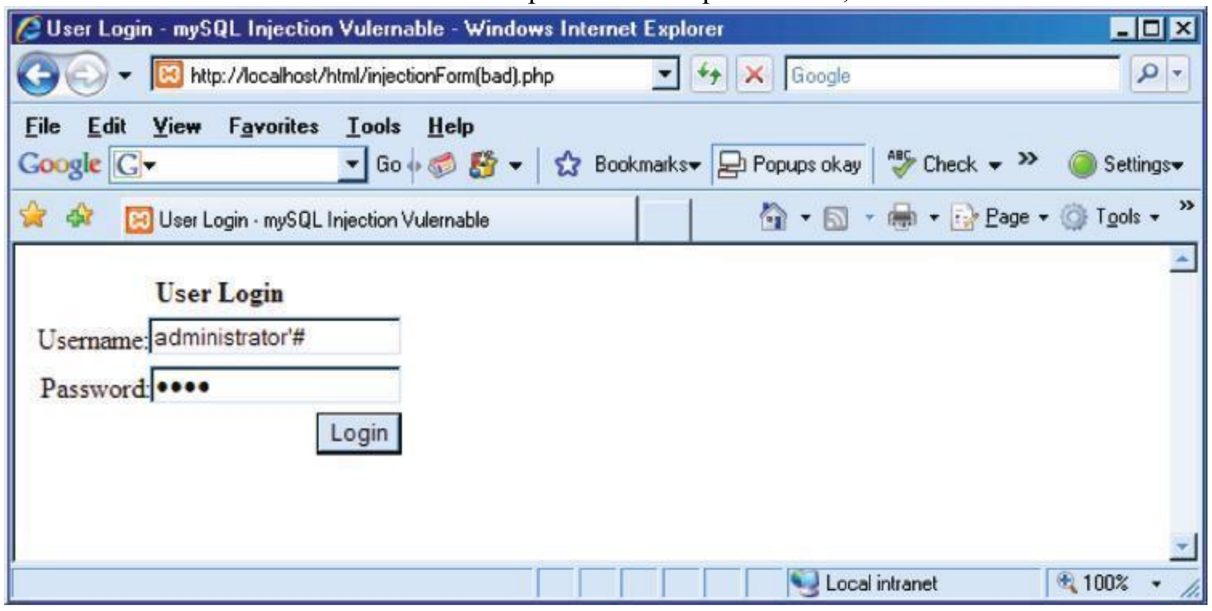
Password:

Login

FIGURE(a) A SQL injection attack.

The attacker wants to take over the administrative privilege of the database and therefore uses the username: administrator'#, as indicated in Figure (b). The # sign indicates the start of a line comment, which although generally useful can typically be ignored. The password can be anything, since the server will ignore anything that follows the # sign. The form of the query and the ignored comment, indicated by the strikethrough, are then

$query = "SELECT username,password FROM login WHERE
username = 'administrator'# AND password = '$password'";


User Login - mySQL Injection Vulernable - Windows Internet Explorer
http://localhost/html/injectionForm(bad).php
Google
File Edit View Favorites Tools Help
Bookmarks
Popups okay
Check
Settings
User Login - mySQL Injection Vulernable
Page
Tools
Local intranet
100%

User Login

Username: administrator'#

Password: ••••

Login

FIGURE(b) The attacker employs the user name: administrator' #.

SQL INJECTION DEFENSE TECHNIQUES:

SQL injection can be protected by filtering the query to eliminate malicious syntax, which involves the employment of some tools in order to

a) scan the source code using, e.g., Microsoft SQL Source Code Analysis Tool,
b) scan the URL using e.g., Microsoft UrlScan,
c) scan the whole site using e.g., HP Scrawlr, and
d) sanitize user input forms through secure programming.

Buffer Overflow:

A buffer overflow occurs when a program or process tries to store more data in a buffer (temporary data Storage area) than it was intended to hold. Since buffers are created to contain a finite amount of data, the extra information - which has to go somewhere - can overflow into adjacent buffers, corrupting or overwriting the valid data held in them. It may occur accidentally through programming error; buffer overflow is an increasingly common type of security attack on data integrity.

In buffer Overflow attacks, the extra data may contain codes designed to trigger specific actions, in effect sending new instructions to the attacked computer that could, for example, damage the user's files, change data, or disclose confidential information. Buffer overflow attacks are said to have arisen because the C programming language supplied the framework, and poor programming practice supplied the vulnerability. Vulnerability to buffer overflow attack was discovered in Microsoft Outlook and Outlook Express. A programming flow made it possible for an attacker to compromise the integrity of the target computer by simply it sending an e-mail message.

Unlike the typical e-mail virus, users could not protect themselves by not Opening. attached files; in fact, the user did not even have to open the message to enable the attack. The programs message header mechanisms had a defect that made it possible for senders to overflow the area with extraneous data, which allowed them to execute whatever type of code they desired on the recipient's computers. Because the process was activated as soon as the recipient downloaded the message from the server, this type of buffer overflow attack was very difficult to defend. Microsoft has since created a patch to eliminate the vulnerability. Buffer overflow vulnerabilities are one of the most common vulnerabilities. These kinds of vulnerabilities are perfect for remote access attacks because they give the attacker a great opportunity to launch and execute their attack code on the target computer. A buffer overflow attack occurs when the attacker intentionally enters more data than a program was written to handle. The data runs over and overflows the section of memory that was set aside to accept it. The extra data overwrites on top on another portion of memory that was meant to hold something else, like part of the program's instructions. This allows an attacker to overwrite data that controls the program and can takeover control of the program to execute the attacker's code instead of the program. In exploiting the buffer overflow vulnerability, the main objective is to overwrite some control information in order to change the flow of control in the program. The usual way of taking advantages of this is to modify the control information to give authority to code provided by the attacker to take control.

The stack is a section of memory used for temporary storage of information. In a stack -based buffer overflow attack, the attacker adds more data than expected to the stack, overwriting data. For example, "Let's say that a program is executing and reaches the stage where it expects to use a postal coder or zip code, which it gets from a Web-based form that customers filled Out. " The longest postal code is fewer than twelve characters, but on the web form, the attacker typed in the letter "A" 256 times, followed by Some other commands. The data overflows the buffer allotted for the zip code and the attacker's commands fall into the stack. After a function is called, the address of the instruction following the function call is pushed onto the stack to be saved so that the function knows where to return control when it is finished.

A buffer overflow allows the attacker to change the return address of a function to a point in memory where they have already inserted executable code. Then control can be transferred to the malicious attack code contained With the buffer, called the payload. The payload is normally a command to allow remote access or some other command that would get the attacker closer to having control of the system. The best defense against any of these attacks is to have perfect programs. In ideal circumstances. every input in every program would do bounds checks to allow only a given number of characters. The refore, the best way to deal with buffer overflow problems is to not allow them to occur in the first place.

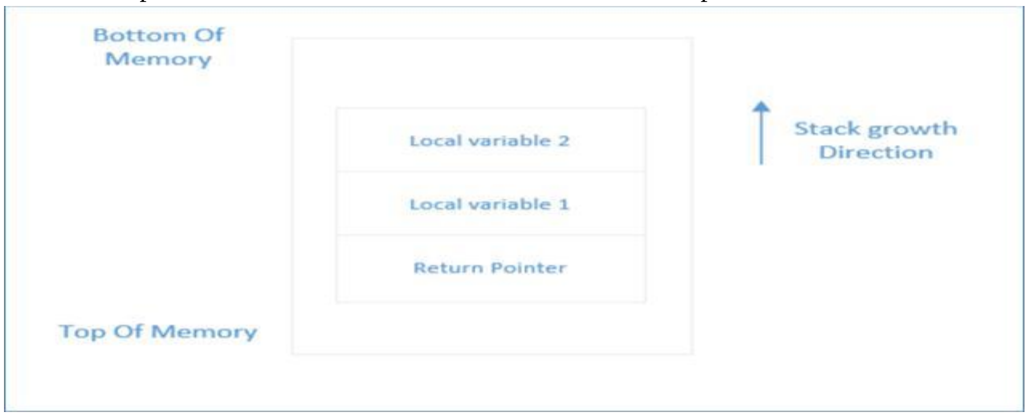


Figure : stack representation of a normal stack

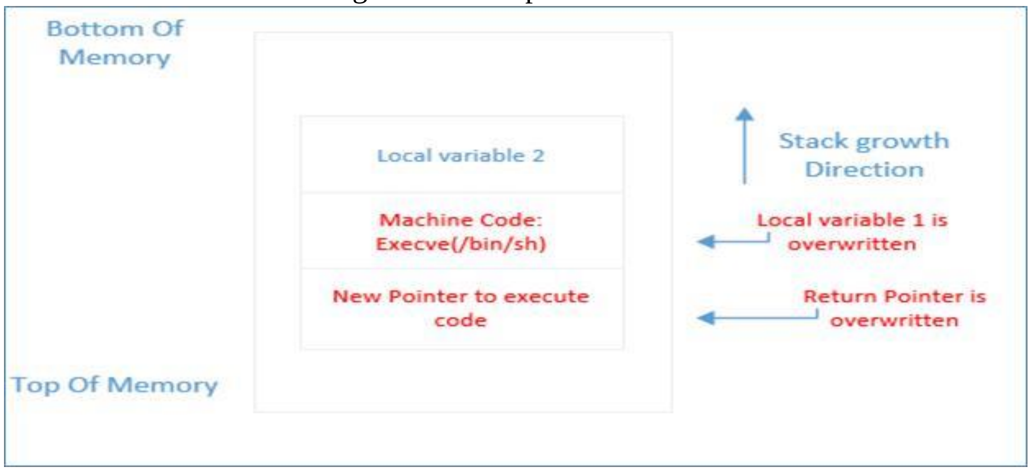

Bottom Of Memory
Local variable 2
Machine Code: Execve(/bin/sh)
New Pointer to execute code
Top Of Memory
Stack growth Direction
Local variable 1 is overwritten
Return Pointer is overwritten

Figure: Buffer Overflow Stack

Buffer Overflow attacks defenses:

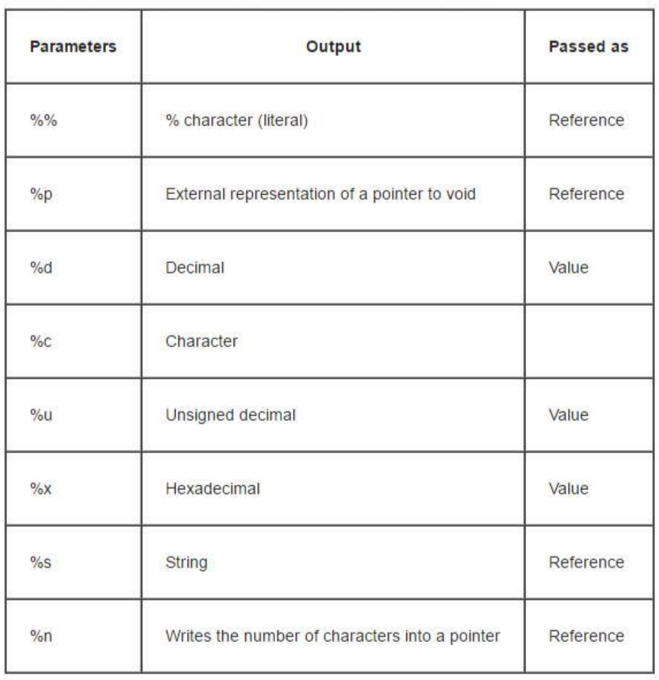
- Check input size wherever applicable, and truncate if it’s too big.
- During the build, make sure that the non-executable system stack is implemented. Stacks are used to store function call arguments, return parameters, local variables but not executable code. So if we can implement a stack which is non-executable stack, a majority of buffer overflow attacks can be controlled. To implement this feature, windows even have a feature called “Data Execution Prevention” which is used to make stack non-executable. DEP settings are available at Systems >Advanced>Performance >Settings>DEP.

Format String:

Buffer overflows aren't the only type of bug that can control a process. Another fairly common programming error is the situation in which a user can control the format parameter to a function, such as printf () or syslog (). These functions take a format string as a parameter that describes how the other parameters should be interpreted. For example, the string specifies that a parameter should be displayed as a signed decimal integer, while %s specifies that a parameter should be displayed as an ASCII string. Format strings give you a lot of control over how data is to be interpreted, and this control can sometimes be abused to read and write memory in arbitrary locations.

To take advantage of format string vulnerability, an attacker gets a computer to display a string of text characters with formatting commands. By carefully manipulating the formatting commands, the attacker can trick the computer into running a program. "Format string bugs are the new trend in computer security vulnerabilities." In the C programming language there are a number of functions which accept a format string as an argument. These functions include fprintf, printf, sprintf, snprintf, vfprintf, vprintf, vsprintf, vsnprintf, setproctitle, syslog and others.

Table 2. Common parameters used in a Format String Attack.



Parameters	Output	Passed as
%%	% character (literal)	Reference
%p	External representation of a pointer to void	Reference
%d	Decimal	Value
%c	Character	
%u	Unsigned decimal	Value
%x	Hexadecimal	Value
%s	String	Reference
%n	Writes the number of characters into a pointer	Reference

Format String Vulnerability Attacks:

Format string vulnerability attacks fall into three categories :

a) Denial Of service
b) Reading
c) Writing

- Format string vulnerability denial of service attacks are characterized by utilizing multiple instances of the %s format specifier to read data off of the stack until the program attempts to read data from an illegal address, which will cause the program to crash.

  printf (userName);

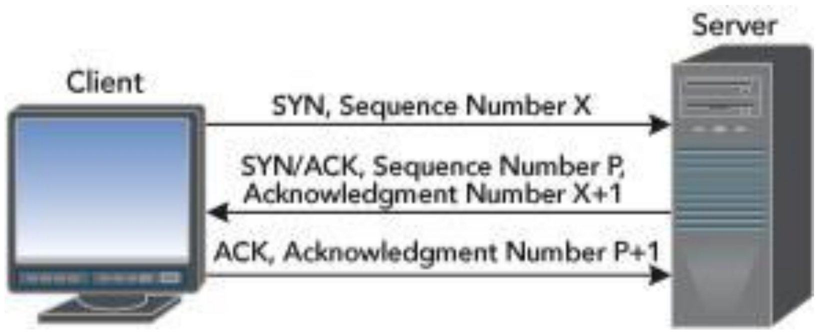
- The attacker could insert a sequence of format strings, making the program show the memory address where a lot of other data are stored, then, the attacker increases the possibility that the program will read an illegal address, crashing the program and causing its non-availability.

  printf ("%s%s%s%s%s%s%s%s%s%s%s%s");

- Format string vulnerability reading attacks typically utilize the %x format specifier to print sections of memory that do not normally have access to.
- Format string vulnerability writing attacks utilize the %d, %u or %x format specifiers to overwrite the Instruction Pointer and force execution of user-supplied shell code.

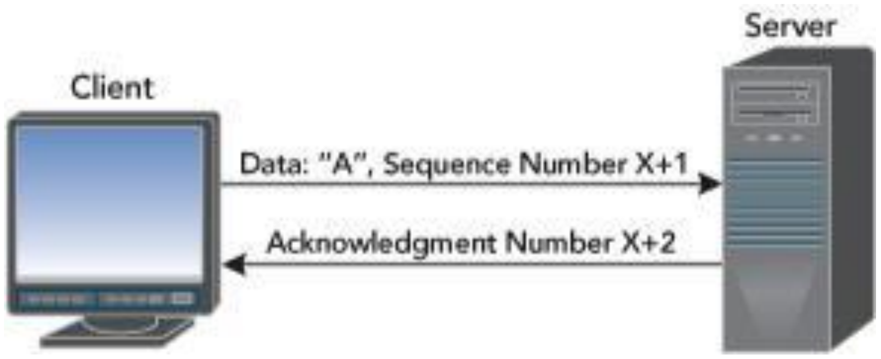
TCP session hijacking:

TCP guarantees delivery of data and also guarantees that packets will be delivered in the same order in which they were sent. In order to guarantee that packets are delivered in the right order, TCP uses acknowledgement (ACK) packets and sequence numbers to create a “full duplex reliable stream connection between two end points,” with the end points referring to the communicating hosts. The connection between the client and the server begins with a three -way handshake.


Server
Client
SYN, Sequence Number X
SYN/ACK, Sequence Number P,
Acknowledgment Number X+1
ACK, Acknowledgment Number P+1

Figure(a): TCP Three-Way Handshake

For now, observe what happens to these sequence numbers when the client starts sending data to the server (see **Figure (b)**). In order to keep the example simple, the client sends the character A in a single packet to the server.



Figure(b) Sending Data over TCP

TCP Session hijacking is when a hacker takes over a TCP session between two machines. Since most authentications only occur at the start of a TCP session, this allows the hacker to gain access to a machine.

A popular method is using source-routed IP packets. This allows a hacker at point A on the network to participate in a conversation between B and C by encouraging the IP packets to pass through its machine. If source-routing is turned Off, the hacker can use “blind" hijacking see figure (c), whereby it guesses the responses of the two machines. Thus, the hacker can send a command, but can never see the However, a common command would be to set a password allowing access from somewhere else on the net. A hacker can also be "inline" between B and C using a sniffing program to watch the conversation. This is known as a "**man-in-the-middle attack**".

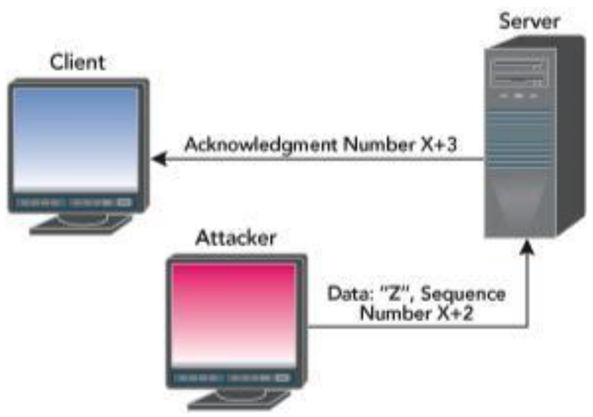


Figure (c) Blind hijacking

Acommon component of such an attack is to execute a denial-of-service attack against one end-point to stop it from responding. This attack can be either against the machine to force it to crash, or against the network connection to force heavy packet loss. TCP session hijacking is a much more complex and difficult attack.

Route Table Modification:

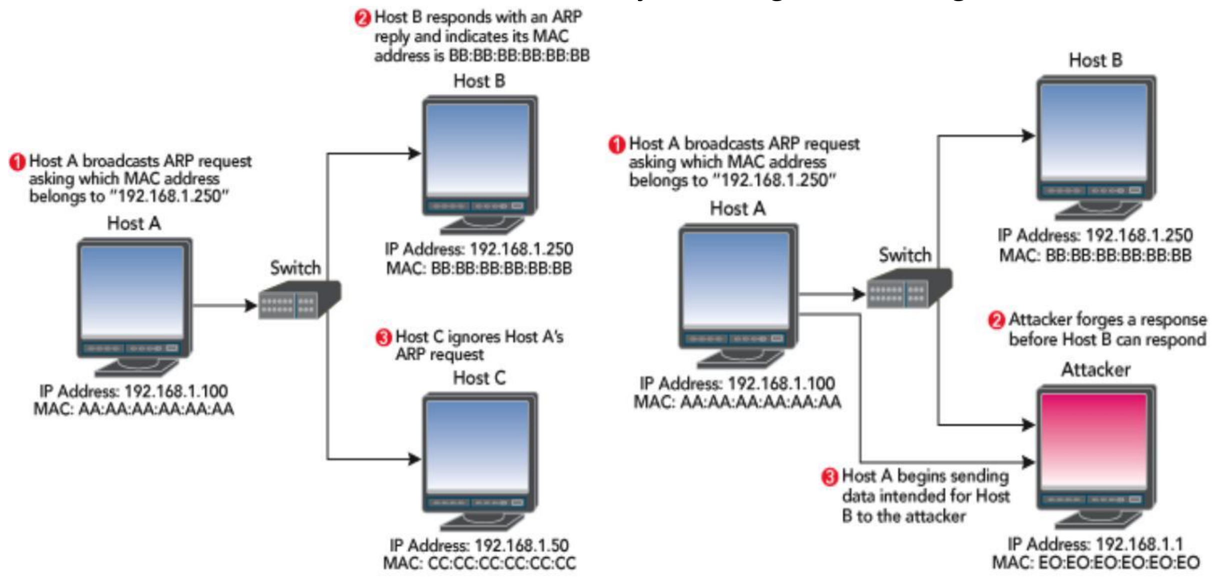
An attacker would be able to put himself in such a position to block packets by modifying routing tables, so that packets flow through a system he has control by changing bridge tables by playing games with spanning-tree frames, or by rerouting physical cables so that the frames must flow through the attacker's system. Most of the time, an attacker will try to change route tables remotely. A more locally workable attack might be to spoof Internet Control Message Protocol (ICMP) and redirect packets to fool some hosts into thinking that there is a better route via the attacker's IP address. Many OS's accept ICMP redirects in their default configuration. Unless, the connection is to be broken entirely (or proxy it in some way), the packets have to be forwarded back to the real router, so they can reach their ultimate destination.

If the attacker has managed to change route tables to get packets to flow through his system, some of the intermediate routers will be aware of the route change, either because of route tables changing or possibly because of an Address Resolution Protocol (ARP) table change. The end nodes would not normally be knowledgeable to this information, if there are at least a few routers between the two nodes. Possibly the nodes could discover the change via a traceroute-style utility, unless the attacker has planned for that and programmed his "router" to account for it.

ARP Attacks:

Another way to make sure that your attacking machine gets all the packets going through it is to modify the ARP tables on the victim machine(s). The address resolution protocol is used by each host on an IP network to map local IP addresses to hardware addresses or MAC addresses. ARP is designed to be a dynamic protocol, so as new machines are added to a network or existing machines get new MAC addresses for whatever reason, the rest update automatically in a relatively short period of time. There is absolutely no authentication in this protocol. Address Resolution Protocol (ARP) spoofing, also known as ARP poisoning or ARP Poison Routing (APR), is a technique used to attack an Ethernet wired or wireless network.

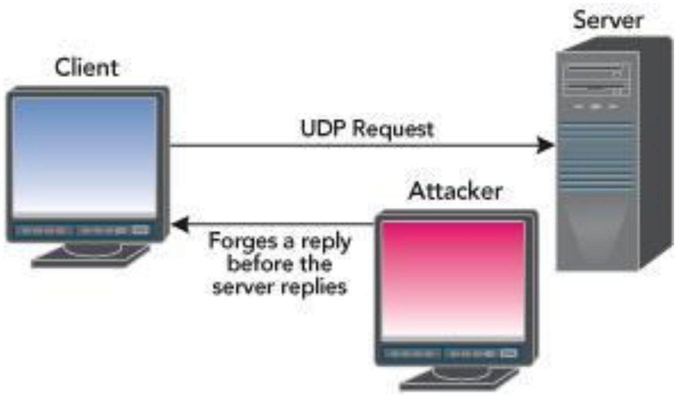
ARP Spoofing allows an attacker to sniff data frames on a local area network (LAN), modify the traffic, or stop the traffic altogether. The attack can only be used on networks that actually make use of ARP and not another method of address resolution. The principle of ARP spoofing is to send fake, or "spoofed", ARP messages to an Ethernet LAN. Generally, the aim is to associate the attacker's MAC address with the IP address of another node. Any traffic meant for that IP address would be mistakenly sent to the attacker instead. The attacker could then choose to forward the traffic to the actual default gateway or modify the data before forwarding it. The attacker could also launch a denial -of-service attack against a victim by associating a nonexistent MAC address to the IP address of the victim's default gateway. ARP spoofing attacks can be run from a compromised host or from an attacker's machine that is connected directly to the target Ethernet segment.



1 Host A broadcasts ARP request asking which MAC address belongs to "192.168.1.250"
Host A
IP Address: 192.168.1.100
MAC: AA:AA:AA:AA:AA:AA
Switch
2 Host B responds with an ARP reply and indicates its MAC address is BB:BB:BB:BB:BB:BB
Host B
IP Address: 192.168.1.250
MAC: BB:BB:BB:BB:BB:BB
3 Host C ignores Host A's ARP request
Host C
IP Address: 192.168.1.50
MAC: CC:CC:CC:CC:CC:CC
1 Host A broadcasts ARP request asking which MAC address belongs to "192.168.1.250"
Host A
IP Address: 192.168.1.100
MAC: AA:AA:AA:AA:AA:AA
Switch
Host B
IP Address: 192.168.1.250
MAC: BB:BB:BB:BB:BB:BB
2 Attacker forges a response before Host B can respond
Attacker
3 Host A begins sending data intended for Host B to the attacker
IP Address: 192.168.1.1
MAC: EO:EO:EO:EO:EO:EO

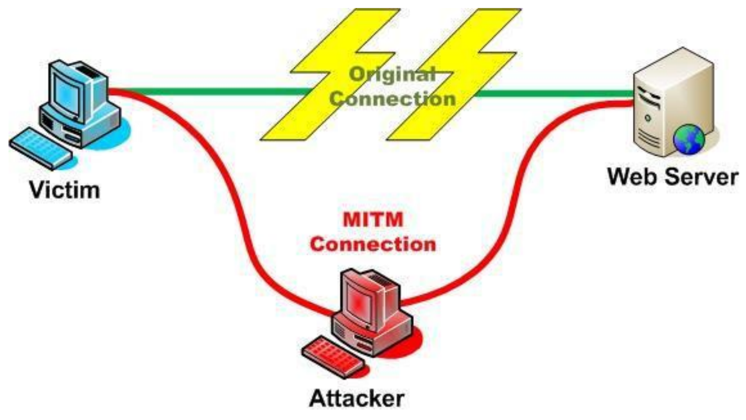
UDP Hijacking:

UDP which stands for User Datagram Protocol is defined as a connectionless protocol. It offers a direct way to send and receive datagram's over an IP network. UDP doesn't use sequence numbers like TCP. It is mainly used for broadcasting messages across the network or for doing DNS queries. Hijacking a session over a User Datagram Protocol (UDP) is exactly the same as over TCP, except that UDP attackers do not have to worry about the overhead of managing sequence numbers and other TCP mechanisms. Since UDP is connectionless, injecting data into a session without being detected is extremely easy.



Man in the Middle Attacks:

In cryptography, a man-in-the-middle attack (MITM) is an attack in which an attacker is able to read, insert and modify at will, messages between two parties without either party knowing that the link between them has been compromised. The attacker must be able to observe and intercept messages going between the two victims.



The MIIM attack may include one or more of

1. Eavesdropping, including traffic analysis and possibly a known-plaintext attack.
2. Chosen ciphertext attack, depending on what the receiver does with a message that it decrypts.
3. Substitution attack
4. Replay attacks
5. Denial of service attack. The attacker may for instance jam all communications before attacking one of the parties. The defense is for both parties to periodically send authenticated status

   messages and to treat their disappearance with paranoia. MITM is typically used to refer to active manipulation Of the messages, rather than passively eavesdropping.

A man-in-the-middle attack can succeed only when the attacker can impersonate each endpoint to their satisfaction as expected from the legitimate other end. Most cryptographic protocols include some form of endpoint authentication specifically to prevent MITM attacks.